Proceedings of the 1968 Proton Linear Accelerator Conference, Upton, New York, USA

PROGRAMS FOR COMPUTER CONTROL OF EPA*

M. D. J. MacRoberts, Sally Ohlsen, and C. M. Plopper,
D. D. Simmonds, and R. F. Thomas

University of California
Los Alamos Scientific Laboratory
Los Alamos, New Mexico

Introduction

One of the purposes to which the Electron Prototype Accelerator at Los Alamos is being put is the development of a prototype computer control system for the LAMPF linear accelerator. A part of this work is the development of a collection of computer programs which will aid in the evaluation of various control techniques and apparatus, identify the critical design parameters of both the programming system and the computer hardware, and eventually serve as a prototype for a programming system for control of LAMPF. In pursuit of these goals about 250 programs with a total of about 50,000 instructions have been written for an SEL-810A computer which has been on the site since February, 1967. Since EPA became operable in December, 1967, we have used this system to conduct a number of exercises in computer control, involving turn-on, turn-off, beam steering and RF control experiments. This paper summarizes the current status of the programming system, illustrates through the description of a few important programs how it is intended to be used, and discusses some current problems and possible solutions to them.

Computer, Peripherals, and Channels

The computer which is being used in this study is an SEL-810A, a machine which has internal registers and memory words sixteen bits in length. Our particular machine has 8192 words of core storage and a magnetic disk with a capacity of about one and a half million words. Two timing figures associated with these devices are relevant to the evaluation of the system as it now stands, to future program design considerations and to future equipment specifications: the average instruction execution time is about 3.5 μsec.; the average time required to obtain data or code from the disk is about .1 sec. The computer is connected to the accelerator through digital interface equipment which is capable of executing a complete control or data acquisition task in approximately one millisecond, unless it is necessary to wait for an accelerator pulse or a mechanical motion. Three hundred and thirty channels of binary data from the accelerator can be read by the computer, as can 133 channels of digitized analog data. Commands are sent to the accelerator by means of 110 binary channels and 59 pulse channels. The accelerator operator's console is also a computer peripheral, consisting of a 9 in. x 9 in. CRT display capable of displaying both characters and vectors, a light pen for interaction directly with displays, a number of status lights, function buttons, toggle switches, and machine readable dials. The speed of operation of this collection of devices runs from microseconds to milliseconds, depending on whether the action is purely electronic or involves mechanical motion. Also in the operating area are a teletype unit and a 4 in. x 4 in. dual trace scope. The teletype is a computer input/output unit; the signals to the scope from the accelerator are directly wired but remotely switchable by the computer. Figure 1 shows the operator's console. Figure 2 shows a schematic diagram of the computer, accelerator, peripheral equipment and communication paths. It should be noted that when operating EPA from the central control room, the operator's only possible path of control is through the computer. There are no hard-wired controls and very little hard-wired data available.

System Programs

Programs which run automatically whenever the operating system is running or whose main purpose is to provide services for other programs will be termed system programs. Their functions include scheduling of other programs, handling of input-output units and responding to priority interrupts. A brief description of some of them should reveal some of the structure of the programming system.

Exec

Half of the core storage of the computer is reserved for permanently resident system programs and tables. The other half of core is used on a temporary basis by all the other programs, whose place of permanent residence is the disk. Programs are not relocatable in core; they must be always executed from the same memory locations. The executive program controls the flow of programs from disk to core, handles requests for programs to be run, either periodically or only once, and switches control of the computer among those programs in core at a given moment. A set of tables containing space for twenty-five programs is used to define the status of all programs which are either running or waiting to run.

When a program is requested to run, its disk location, length and core requirements must first be obtained from a special table on the disk. Then a check is made to see if the required core locations are available. If they are, loading is initiated from disk; if not, no action is taken until they are. When the tables indicate that loading is complete, the program is given control. The program executes until it must wait for some event (usually completion of an input or output function); it then yields control to exec. Exec proceeds to examine each of the slots in the tables in turn, round robin fashion, to see whether the program, if any, occupying that slot can make any further progress. When a program signals that it is finished, its core reservation is cancelled and its space in the tables may be freed. The program is not written back onto disk by exec; however a very few programs perform this function for themselves.

The shared segment of core is partitioned into eight blocks; it is therefore possible to have as many as eight independent strings in simultaneous execution in core at any given time. There is no priority of execution of these programs.

I/O Drivers

Five programs in the core-resident system are devoted to the operation of the disk, CRT display, teletype units, and the computer interface unit. These programs, at the request of other programs, schedule, perform and check transmissions to and from the peripheral units. They contain interrupt servicers for error or end-of-transmission conditions, and they have the duty of informing the requesting program when the transmission is complete.

Interrupt Servicers

Priority interrupts cause an interruption of the normal sequence of instructions and start a program in a new location. Many of these interrupts are associated with input/output units, and the associated I/O drivers contain subprograms to handle them. A few are not connected to peripheral units, and therefore their servicing programs exist independently. A priority interrupt is simply a method of rapidly informing the computer that something has happened. The events are varied, ranging from a teletype key having been pressed to a power failure. Consequently the computer responses are also varied. Their common characteristic is that they must restart the interrupted program so that it will perform as if there had been no interruption; consequently they are denied the use of many useful system functions which it is possible for them to interrupt.

Message Handler

The message handler is a nonresident system program which prints or displays messages at the request of other programs on either of the teletypes, the line printer, or the CRT. Messages are stored in core and on the disk in a standard format which includes a destination code. If the destination unit is not available, the message handler automatically chooses an alternative unit for output. In order to lessen the probability for programs having to wait for slow devices like teletypes, a ten-position queue is maintained, and a program is given a finished signal when its message has been entered in the queue rather than when the transmission actually occurs.

Interpreter

In order to facilitate the writing and testing of accelerator sequencing and control programs a special language has been devised. Programs written in this accelerator language are assembled by the accelerator language assembler and the resulting object code is executed interpretively under the system by the accelerator language interpreter. The interpreter is brought into core whenever an accelerator language program is in operation and remains until there are no more interpretive codes running. The interpreter and the codes which are being interpreted occupy together about 1500 words of core storage during this time. Its functions are to manage storage for the codes it processes (it uses a demand paging scheme to allocate about 500 words of the 1500), to maintain displays, to deal with light pen hits on the display, and to interpret and execute the instructions of the interpretive language.

Application Programs

Programs which can be commanded to run at the discretion of the accelerator operator and whose principal function is to provide services for him will be called application programs. A functional description of the more important ones should provide a picture of the kind of operation toward which we are working.

Data Scan

A fundamental task of the computer is the gathering and recording of data on accelerator operation. The data must also be checked to be sure that various devices are in the correct state. The data scan program provides a comprehensive service of this type. It reads all data channels and stores the data both on the disk and in special core locations reserved for this purpose, where it is easily available to other programs. On request data scan checks to see that on/off data is in the correct state and that analog data falls within prescribed limits. Not all data are necessarily checked. A channel may be added to or removed from the set that is checked either by the operator or by another program. When a channel is found to be wrong, the standard procedure is simply to log the fact and inform the operator, who can then take appropriate action. While this is adequate for current EPA operations, more elaborate analyses of problems will probably be developed for LAMPF so that troubles can be more accurately pinpointed and corrective measures taken either by operating personnel or by the computer.

Data scan is capable of performing its routine task within one second, and the usual mode of operation is for it to run every second. However, the console buttons provide controls by which it can be started, stopped or run one time only at the operator's discretion.

Line-by-Line Display

The second basic task of the computer is to present information about the accelerator to the operator in a convenient form. The primary display device is the CRT, and the most general and most commonly used display is known as line-by-line. By setting dials to channel numbers and pressing function buttons the operator can display up to fifteen different data channels on the face of the CRT. Each channel is identified by number and by an eighteen character label. The on/off status or scaled analog value is displayed next to the label and updated either once per second or four times per second, depending on what else is taking place. A channel which is no longer of interest can be erased with a light pen and a new channel requested to take its place. Since it is rather tedious to dial up a long list of channels, a facility is provided for saving a list on the disk and then calling it back with a minimum of dialing and button-pushing.

The same facilities available through the console are available to other programs and this capability is useful in providing for easy human monitoring of automatic accelerator control sequences. The control program requests the line-by-line display which will permit following its progress.

Console Command Programs

Four programs service the control buttons and switches on the operator's console:

Binary Command. The binary command program reads the module and channel numbers from the console dials, checks them for legality, converts them to the correct form for transmission to the interface hardware, and transmits the requested on or off command.

Analog Command. The analog command program adjusts setpoints and potentiometers to desired settings. It reads the channel identification and desired setpoint from console dials and the current position from a data channel associated with the setting. It then calculates from calibration curves the correct number of pulses to be issued to a pulse motor to make the change. The reading of the data channel is checked after the motion has stopped, and the process is repeated until the target value has been achieved. These repetitions tend to correct for mechanical lash, hysteresis, and coarseness or inaccuracies in the calibration curves.

Slew. The operator's console contains three sets of switches and dials referred to as slew controls. The switches may be raised or lowered and they return automatically to center when released. When a switch is deflected, the console generates interrupts to the computer at a rate of ten per second. The program which services these interrupts then sends pulses to the command channel specified by the dials associated with the switch. The number of pulses sent is such that the channel will move .1% of full scale for each interrupt. While one of the slew switches is deflected the line-by-line display update rate is increased to four times per second rather than the customary once per second, and data is read more frequently on the channel which is being slewed. In this way the operator can follow progress on the CRT with good response.

Video Commands. Another set of console buttons and dials enable the operator to switch signals to the two-trace scope in the control area. The computer plays a minor role in this process, serving only to transmit the commands to the remote switching equipment.

Special Purpose Displays

The programs discussed up to this point serve only to make manual control of the accelerator possible through the computer. They provide functions which might be expected to exist in one form or another at any control panel. The application programs which are described next generally will provide a degree of automation in the operation or else provide some feature which would be expensive or difficult without a computer. The first set of these programs to be described, the special displays, present data in more easily comprehended form than is possible with the line-by-line display.

Vary. This program provides a facility for stepping any command channel through some range and plotting some data channel as a function of that command channel setting. Once given the range and increment, the program issues the commands automatically. There are possibilities for both linear and logarithmic scales, different plotting characters and multiple plots on the same graph. There is no easy way of obtaining good permanent copies of the graphs; however, a rough sketch can be produced on the line printer. Fig. 3 shows a picture of a graph produced by Vary.

Timeplot. Program Timeplot plots the value of any data channel as a function of time. The plotting takes place in real time at intervals specified by the operator. The program is similar to Vary in its plotting capabilities, but no commands are issued, and the abscissa is always time. Figure 4 shows a picture of a graph produced by Timeplot.

Beamplot. The Beamplot program is used to facilitate manual beam steering by the operator. It plots the beam current and the x and y coordinates of the beam position at seven stations along the length of the accelerator. The channels are read and the plot updated four times per second. An associated program provides the operator with the ability to assign the slew switches to different focusing and steering magnets with the light pen. A pointer below the graph indicates which station is being manipulated. The slew switches are connected through the program to the magnet controls for the indicated station. Since the pointer can be moved from one station to another with the light pen, the operator can switch quickly and easily from station to station without the relatively slow and awkward dialing of channel numbers on the slew dials. A picture of the plot is shown in Figure 5.

Automatic Accelerator Procedures

A number of programs have been written to run EPA and its associated power supplies through various standard sequences. To date these have fallen into four categories: turn-on, turn-off, hardware diagnostic, and beam search. The earlier programs of this kind were written in SEL assembly language. Since the beginning of this year, however, they have been written primarily in the new accelerator language, which was invented for this purpose. Although the language is a so-called "low-level" language, being an assembly language for an imaginary computer, it offers a great deal of simplification and economy in the coding of these procedures, and the associated interpreter provides many useful features for program debugging, simultaneous execution of sequences, communication between programs, and operator communication. A detailed description of two programs should illustrate the kind of things that these programs are doing.

Prepare Tanks for R.F. The object of this program is to check the temperature and vacuum in the main accelerator tanks, to bring the temperature to the correct value, and inform the operator if the vacuum is not satisfactory. There is a local temperature controller which will maintain the temperature at a setpoint during operation. There is also a water heater connected to the apparatus which provides a source of hot water to speed the warm-up.

The program first checks the position of two local/remote switches which must be set to remote in order for it to have control. Two water pumps are then turned on, if they are not already on, and the water flow is checked. The mixing tank heaters are checked and turned on if they were off. If the tank copper temperature is more than half a degree below the required value, the hot water valve is opened and left open until the water temperature either comes within half a degree of the setpoint or else begins to drop, indicating that the water heater has been emptied. The program then loops, watching the copper temperature. An hour later if the temperature is still not within half a degree, another slug of hot water is dumped into the system, the water heater having recovered by this time. This process will be repeated indefinitely. In practice two hot water dumps are usually necessary, starting with the tanks cold. At the beginning of the warm-up the vacuum is checked and the operator is notified of the result so that he can take advantage of the long warm-up time to make any necessary repairs or adjustments. At the end of the warm-up the vacuum is checked again, and if it is all right, the program terminates.

Beam Search. At the time of writing, the beam position monitors on EPA are not yet functioning. Consequently the task of this program is to locate the beam and center it as well as it can, using only beam current information. The program assumes that a previous program has turned on all the necessary power supplies and checked the local/remote switches. It is also assumed that there exists a low-intensity beam. Beam Search executes a search pattern, changing the currents in the horizontal and vertical steering magnets in a systematic way in steps sufficiently small so that it will not be in danger of stepping across the beam without seeing it. After each change in the steering magnets the beam current is read. When a current above some threshold is measured, the program attempts to center the beam by finding the cutoff settings on each steering magnet and then setting the currents to the means of the cutoffs. A final procedure checks this setting by attempting to maximize the current reading at the station. Small steps are taken in all directions away from the assumed center position, and if any of these results in an increase in the current, it is taken as the new center position. If no improvement is found, the program is finished with the station.

These two programs are representative of about twenty which have been written to implement turning on and off and equipment testing. Essentially all the parts of EPA which are controllable by the computer now have such programs to service them. Since EPA is an experimental prototype, its parts are run separately on a piecemeal basis as often as it is run as a whole. Consequently there is not yet a program which will bring up the whole facility from a cold start, run it, and then shut it off. Since a program can start up another program, however, it would be an easy matter at this stage to write a master control program which would, by calling the various component sequences, do such a thing.

Experience

During the past four months the basic programming system has undergone only minor modifications while being used extensively for manual operation and for developing automatic sequencing programs. Observation of the performance of the system during this time has indicated no major flaws in the general design, success in performing several tasks in parallel, and some problems in speed of response and system reliability. The problems which have turned up thus far appear to be soluble in straightforward ways.

As was mentioned earlier, half of the core storage of the computer is devoted to system programs and tables which are always in core. The other half is again halved and the resulting blocks assigned to two categories of programs which are not always in core. About 2000 words are assigned to programs which occupy core for long periods of time such as the interpreter, slew, and certain turn-on sequences. The remaining 2000 words are used by programs which run rapidly to completion. The degree of concurrent operation which is possible under this scheme is considerable. As an example, the system can have up to ten automatic sequences running under the interpreter, run the data scan program, record data, maintain the line-by-line display, type messages, and still be responsive to commands from the operator's console.

In this fairly busy state it might well take the computer three or four seconds to respond to a console command. This response is slow enough to be unsettling to an operator and to cause him to wonder whether he is really in control. This sluggishness is caused by the large number of disk accesses required when the system is busy combined with the relatively slow access time of the disk. Three remedies appear to be available: (1) the programs and data have not been organized in such a way as to minimize the number of disk transfers nor to maximize the efficiency of those which take place. There may be a factor of two or three available from this source. (2) Larger core storage would reduce the traffic in programs and consequently the number of disk transfers. (3) A drum or head-per-track disk might reduce the average time per transfer by a factor of ten or so.

Our second major problem has been that of recovery from hardware malfunction. The system as currently organized has a tendency to expect everything to work perfectly, and consequently when some expected event fails to occur, a log-jam rapidly develops, in that most of the core will soon be reserved by programs which are, for one reason or another, waiting for the lost event. The answer to this problem is to set time limits on expected events. If the event does not occur within the time allotted, remedial action will be taken. This approach has been successfully implemented in some programs and it is a matter of applying such fault-tolerant techniques universally to greatly improve the system reliability.

Because each nonresident program can be loaded only into its own core locations, it is quite possible to have some incompatible program configurations. These are quickly detected and so far we have been able to assign core locations in such a way as to obtain a working set of programs which call each other. This limitation restricts the combinations of long-running programs which can run in parallel, however, and complicates the organization of all programs. Relocation of programs would make the system more general and easier to use.

Conclusion

The application programs which have been described in this paper, and which have been written so far, perform just three tasks: they enable an operator to exercise control over the accelerator from the control room via the computer, they record data from the accelerator, and they imitate the actions of operators in performing routine procedures such as turn-on sequences. The first two tasks are vital elements, the third a desirable element of computer control, and they form a sound basis upon which one may build more elaborate services. The routine operator tasks have turned out to result in programs which are not especially complex, when stated in appropriate terms, since they are able to call upon many other, low-level, programs to perform significant parts of their jobs. The EPA goal has been to construct a quite general system upon a small, limited computer. The general approach to computer control which has been taken in this case appears to be valid, and the experience has been valuable in indicating the critical features of a computer intended to control LAMPF and of its associated programming system.

*Work performed under the auspices of the U. S. Atomic Energy Commission.

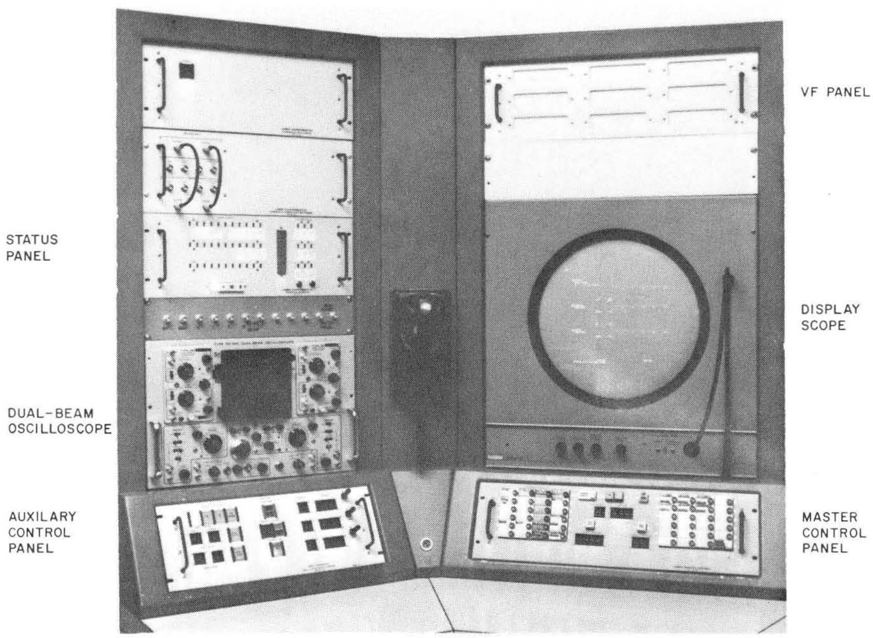

Figure 1. EPA Control Console

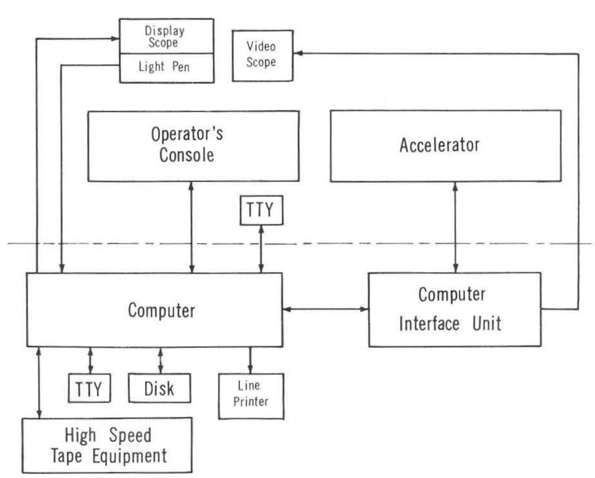

Figure 2. EPA and Computer System. The accelerator operator is concerned primarily with the units in the upper half of the drawing.

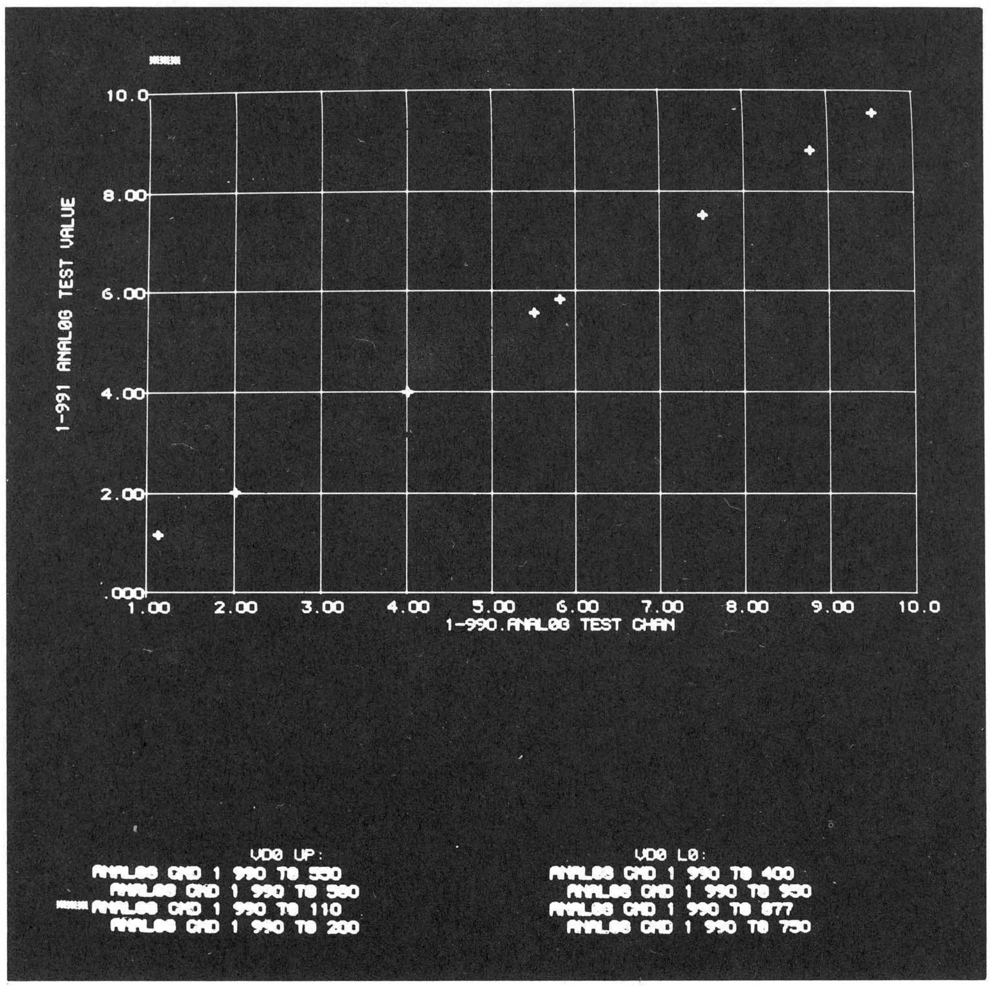

Figure 3. Vary Plot.

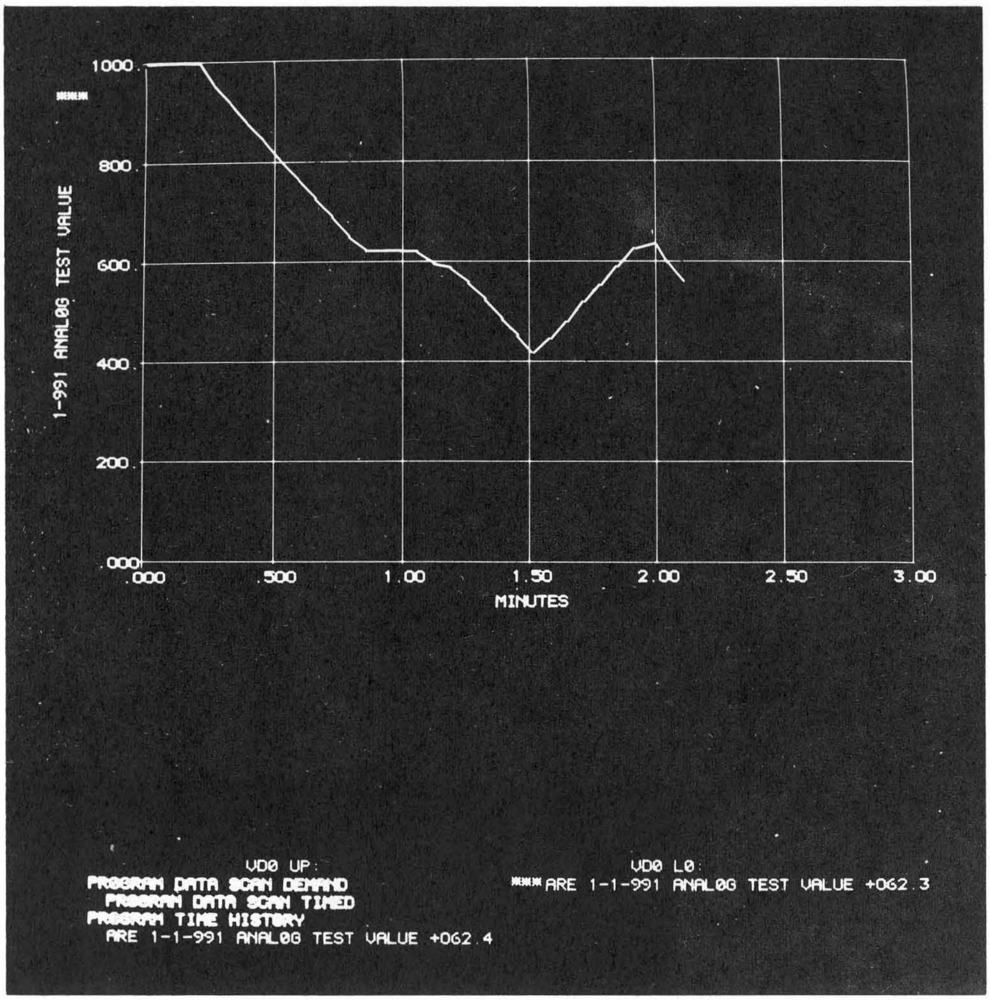

Figure 4. Timeplot.

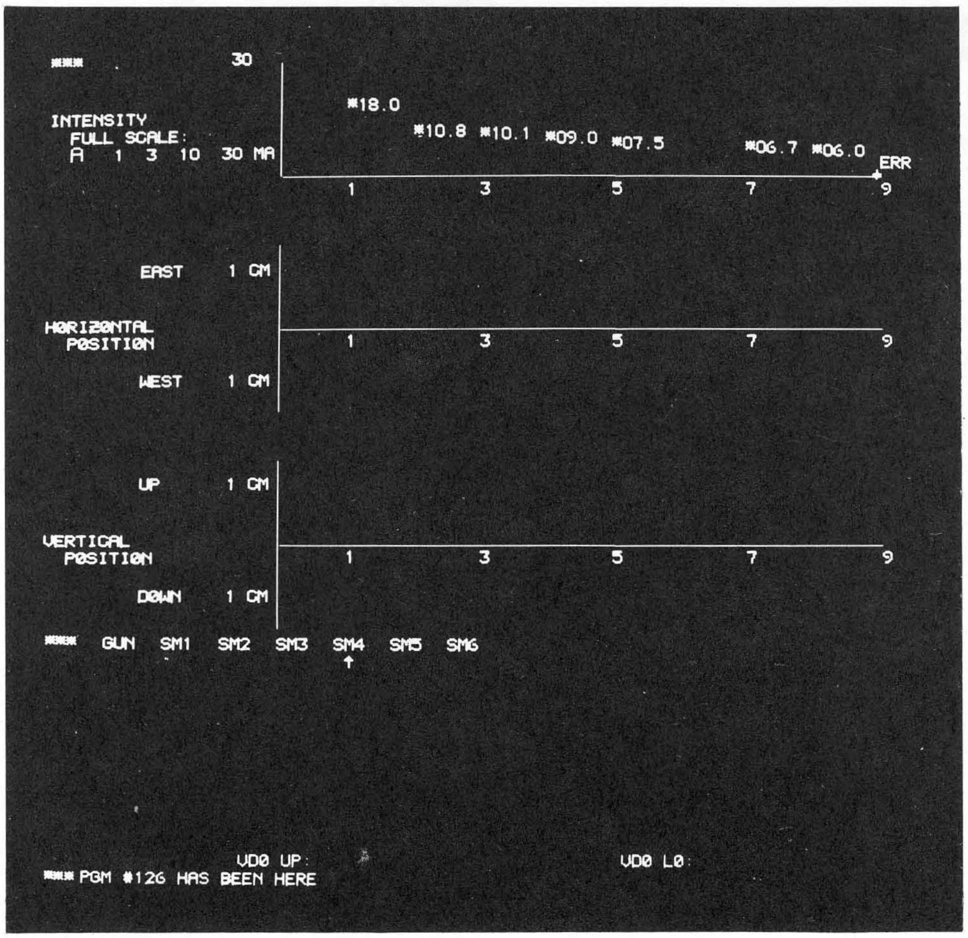

Figure 5. Beamplot display.